PHPUnit to the limit

International PHP Conference 2010

Tobias Schlitt <toby@qafoo.com>
Kore Nordmann <kore@qafoo.com>

October 11, 2010

Outline

About us

- Degree in computer sience in 2010
- Each more than 10 years of professional PHP
- Open source enthusiasts
- Contributing to various FLOSS projects
- 2 of 3 founders of **Qafoo GmbH**, which provides **Services all around high quality PHP**

PHPUnit

Unit Testing Framework

PHPUnit

Unit Testing Framework

- Unit tests
- Test doubles
- Code coverage
- Skeleton generation

PHPUnit

~~Unit~~ Testing Framework

- Unit tests
- Test doubles
- Code coverage
- Skeleton generation
- DB test extension
- Selenium integration
- . . . let's see . . .

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . has the "1 assertion per test" paradigm?
- . . . has a solution?
- . . . has UnderControll?
- . . . has Selenium?
- . . . has Clover, Bamboo?
- . . . has anything else?

Quality survey

Who of you . . .

- . . . uses PHPUnit?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . sticks to the “1 assertion per test” paradigm?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . sticks to the “1 assertion per test” paradigm?
- . . . uses a CI solution?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . sticks to the “1 assertion per test” paradigm?
- . . . uses a CI solution?
  - phpUnderControll?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . sticks to the “1 assertion per test” paradigm?
- . . . uses a CI solution?
  - phpUnderControll?
  - Hudson?
  - Bamboo?
  - anything else?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . sticks to the “1 assertion per test” paradigm?
- . . . uses a CI solution?
  - phpUnderControll?
  - Hudson?
  - Atlassian Bamboo?
  - anything else?

Quality survey

Who of you . . .

- . . . uses PHPUnit?
- . . . uses PHPUnit in *strange* ways?
- . . . sticks to the “1 assertion per test” paradigm?
- . . . uses a CI solution?
  - phpUnderControll?
  - Hudson?
  - Atlassian Bamboo?
  - anything else?

Outline

Outline

Testing file system access

- Abstract file system access
- Allows mocking
- Still, back end needs to be tested
- Can be cumbersome and sometimes hard

Testing file system access

- Abstract file system access
- Allows mocking
- Still, back end needs to be tested
- Can be cumbersome and sometimes hard
- vfsStream to the rescue
  - `http://bit.ly/vfsStream`

```
1 $ pear channel-discover pear.php-tools.net
2 $ pear install pat/vfsStream-beta
```

Simple logger class

```
1  class qaLogger
2  {
3      public function __construct( $logFile, $verbosity )
4      {
5          $this->logFile   = $logFile;
6          $this->verbosity = $verbosity;
7          $this->initFileDescriptor();
8      }
```

Simple logger class

```
1     private function initFileDescriptor()
2     {
3         if ( false === $this->canWriteLogFile( $this->logFile ) )
4         {
5             throw new RuntimeException( "Cannot write to log file '{$this->
                  logFile}'." );
6         }
7
8         $this->fileDescriptor = fopen( $this->logFile, 'a' );
9
10        if ( false === $this->fileDescriptor )
11        {
12            throw new RuntimeException( "Error while opening log file '{$this->
                  logFile}'." );
13        }
14    }
```

Simple logger class

```
1   private function initFileDescriptor()
2   {
3       if ( false === $this->canWriteLogFile( $this->logFile ) )
4       {
5           throw new RuntimeException( "Cannot write to log file '{$this->
                logFile}'." );
6       }
7
8       $this->fileDescriptor = fopen( $this->logFile, 'a' );
9
10      if ( false === $this->fileDescriptor )
11      {
12          throw new RuntimeException( "Error while opening log file '{$this->
                logFile}'." );
13      }
14  }
```

Simple logger class

```
1     private function canWriteLogFile( $logFile )
2     {
3         return (
4             file_exists( $logFile ) && is_writable( $logFile )
5             || !file_exists( $logFile ) && is_writable( dirname( $logFile ) )
6         );
7     }
```

Simple logger class

```
1   public function log( $text, $type )
2   {
3       if ( !( $this->verbosity & $type ) )
4       {
5           return;
6       }
7
8       fwrite(
9           $this->fileDescriptor,
10          sprintf(
11              "%s (%s):\n",
12              $this->typeNameMap[$type],
13              date( 'r' )
14          )
15      );
16      fwrite( $this->fileDescriptor, $text );
17      fwrite( $this->fileDescriptor, "\n" );
18  }
```

Simple logger class

```
1     public function __destruct()
2     {
3         if ( is_resource( $this->fileDescriptor ) )
4         {
5             fclose( $this->fileDescriptor );
6         }
7     }
8 }
```

Setting up vfsStream

```
1  class qaLoggerTest extends PHPUnit_Framework_TestCase
2  {
3      const ROOT_DIR = 'root';
4
5      protected $rootDir;
6
7      protected function setUp()
8      {
9          if ( !class_exists( 'vfsStream' )
10             && false === @include_once( 'vfsStream/vfsStream.php' ) )
11         {
12             $this->markTestSkipped( 'Missing vfsStream.' );
13         }
14         vfsStreamWrapper::register();
15
16         $this->rootDir = vfsStream::newDirectory( self::ROOT_DIR );
17         vfsStreamWrapper::setRoot( $this->rootDir );
18     }
```

Setting up vfsStream

```
1  class qaLoggerTest extends PHPUnit_Framework_TestCase
2  {
3      const ROOT_DIR = 'root';
4
5      protected $rootDir;
6
7      protected function setUp()
8      {
9          if ( !class_exists( 'vfsStream' )
10             && false === @include_once( 'vfsStream/vfsStream.php' ) )
11         {
12             $this->markTestSkipped( 'Missing vfsStream.' );
13         }
14         vfsStreamWrapper::register();
15
16         $this->rootDir = vfsStream::newDirectory( self::ROOT_DIR );
17         vfsStreamWrapper::setRoot( $this->rootDir );
18     }
```

Setting up vfsStream

```
1  class qaLoggerTest extends PHPUnit_Framework_TestCase
2  {
3      const ROOT_DIR = 'root';
4
5      protected $rootDir;
6
7      protected function setUp()
8      {
9          if ( !class_exists( 'vfsStream' )
10             && false === @include_once( 'vfsStream/vfsStream.php' ) )
11         {
12             $this->markTestSkipped( 'Missing vfsStream.' );
13         }
14         vfsStreamWrapper::register();
15
16         $this->rootDir = vfsStream::newDirectory( self::ROOT_DIR );
17         vfsStreamWrapper::setRoot( $this->rootDir );
18     }
```

Failure testing

```
1     /**
2      * @expectedException RuntimeException
3      */
4     public function testCtorFailureDirNotWritable()
5     {
6         // Not writable dir
7         $logDir = vfsStream::newDirectory( 'log', 0 );
8         $this->rootDir->addChild( $logDir );
9
10        $logger = new qaLogger(
11            vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
12            qaLogger::TYPE_NOTHING
13        );
14    }
```

Failure testing

```
1     /**
2      * @expectedException RuntimeException
3      */
4     public function testCtorFailureDirNotWritable()
5     {
6         // Not writable dir
7         $logDir = vfsStream::newDirectory( 'log', 0 );
8         $this->rootDir->addChild( $logDir );
9
10        $logger = new qaLogger(
11            vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
12            qaLogger::TYPE_NOTHING
13        );
14    }
```

Failure testing

```
1     /**
2      * @expectedException RuntimeException
3      */
4     public function testCtorFailureDirNotWritable()
5     {
6         // Not writable dir
7         $logDir = vfsStream::newDirectory( 'log', 0 );
8         $this->rootDir->addChild( $logDir );
9
10        $logger = new qaLogger(
11            vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
12            qaLogger::TYPE_NOTHING
13        );
14    }
```

Failure testing

```
1     /**
2      * @expectedException RuntimeException
3      */
4     public function testCtorFailureFileNotWritable()
5     {
6         $logDir  = vfsStream::newDirectory( 'log', 0777 );
7         $this->rootDir->addChild( $logDir );
8
9         $logFile = vfsStream::newFile( 'log.file', 0 );
10        $logDir->addChild( $logFile );
11
12        $logger = new qaLogger(
13            vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
14            qaLogger::TYPE_NOTHING
15        );
16    }
```

Failure testing

```
1     /**
2      * @expectedException RuntimeException
3      */
4     public function testCtorFailureFileNotWritable()
5     {
6         $logDir  = vfsStream::newDirectory( 'log', 0777 );
7         $this->rootDir->addChild( $logDir );
8
9         $logFile = vfsStream::newFile( 'log.file', 0 );
10        $logDir->addChild( $logFile );
11
12        $logger = new qaLogger(
13            vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
14            qaLogger::TYPE_NOTHING
15        );
16    }
```

File creation test

```
1   public function testCtorSuccessNewLog()
2   {
3       $logDir = vfsStream::newDirectory( 'log', 0777 );
4       $this->rootDir->addChild( $logDir );
5
6       $logger = new qaLogger(
7           vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
8           qaLogger::TYPE_NOTHING
9       );
10
11      $this->assertEquals(
12          1,
13          count( $logDir->getChildren() )
14      );
15
16      $this->assertAttributeType(
17          'resource',
18          'fileDescriptor',
19          $logger
20      );
21  }
```

File creation test

```
1   public function testCtorSuccessNewLog()
2   {
3       $logDir = vfsStream::newDirectory( 'log', 0777 );
4       $this->rootDir->addChild( $logDir );
5
6       $logger = new qaLogger(
7           vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
8           qaLogger::TYPE_NOTHING
9       );
10
11      $this->assertEquals(
12          1,
13          count( $logDir->getChildren() )
14      );
15
16      $this->assertAttributeType(
17          'resource',
18          'fileDescriptor',
19          $logger
20      );
21  }
```

File creation test

```
1   public function testCtorSuccessNewLog()
2   {
3       $logDir = vfsStream::newDirectory( 'log', 0777 );
4       $this->rootDir->addChild( $logDir );
5
6       $logger = new qaLogger(
7           vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
8           qaLogger::TYPE_NOTHING
9       );
10
11      $this->assertEquals(
12          1,
13          count( $logDir->getChildren() )
14      );
15
16      $this->assertAttributeType(
17          'resource',
18          'fileDescriptor',
19          $logger
20      );
21  }
```

File creation test

```
1   public function testCtorSuccessNewLog()
2   {
3       $logDir = vfsStream::newDirectory( 'log', 0777 );
4       $this->rootDir->addChild( $logDir );
5
6       $logger = new qaLogger(
7           vfsStream::url( self::ROOT_DIR . '/log/log.file' ),
8           qaLogger::TYPE_NOTHING
9       );
10
11      $this->assertEquals(
12          1,
13          count( $logDir->getChildren() )
14      );
15
16      $this->assertAttributeType(
17          'resource',
18          'fileDescriptor',
19          $logger
20      );
21  }
```

File writing test

```
1   public function testLogWritten()
2   {
3       $logger = new qaLogger(
4           vfsStream::url( self::ROOT_DIR . '/log.file' ),
5           qaLogger::TYPE_REQUEST
6       );
7
8       $logger->log(
9           'Some_log_text',
10          qaLogger::TYPE_REQUEST
11      );
12
13      $logDirChildren = $this->rootDir->getChildren();
14      $this->assertEquals(
15          1,
16          count( $logDirChildren )
17      );
18      $this->assertGreaterThan(
19          15,
20          strlen( $logDirChildren[0]->getContent() )
21      );
22  }
```

File writing test

```
1   public function testLogWritten()
2   {
3       $logger = new qaLogger(
4           vfsStream::url( self::ROOT_DIR . '/log.file' ),
5           qaLogger::TYPE_REQUEST
6       );
7
8       $logger->log(
9           'Some log text',
10          qaLogger::TYPE_REQUEST
11      );
12
13      $logDirChildren = $this->rootDir->getChildren();
14      $this->assertEquals(
15          1,
16          count( $logDirChildren )
17      );
18      $this->assertGreaterThan(
19          15,
20          strlen( $logDirChildren[0]->getContent() )
21      );
22  }
```

File writing test

```
1   public function testLogWritten()
2   {
3       $logger = new qaLogger(
4           vfsStream::url( self::ROOT_DIR . '/log.file' ),
5           qaLogger::TYPE_REQUEST
6       );
7
8       $logger->log(
9           'Some log text',
10          qaLogger::TYPE_REQUEST
11      );
12
13      $logDirChildren = $this->rootDir->getChildren();
14      $this->assertEquals(
15          1,
16          count( $logDirChildren )
17      );
18      $this->assertGreaterThan(
19          15,
20          strlen( $logDirChildren[0]->getContent() )
21      );
22  }
```

File writing test

```
1   public function testLogWritten()
2   {
3       $logger = new qaLogger(
4           vfsStream::url( self::ROOT_DIR . '/log.file' ),
5           qaLogger::TYPE_REQUEST
6       );
7
8       $logger->log(
9           'Some log text',
10          qaLogger::TYPE_REQUEST
11      );
12
13      $logDirChildren = $this->rootDir->getChildren();
14      $this->assertEquals(
15          1,
16          count( $logDirChildren )
17      );
18      $this->assertGreaterThan(
19          15,
20          strlen( $logDirChildren[0]->getContent() )
21      );
22  }
```

Outline

Testing image generation

- Images are binary data
- Binary data generation “cannot be tested”
- Abstract it away as far as possible / sensible

Ultimately those “drivers” need to be tested, too.

Testing image generation

- Images are binary data
- Binary data generation “cannot be tested”
- Abstract it away as far as possible / sensible
- But eventually those “drivers” need to be tested, too.

Use case

- Graph component from Apache Zeta Components (formerlly eZ Components)
  - Renders charts using PHP
  - Test driven development
  - Implements different drivers: GD (PNG, JPEG), Flash, SVG, Cairo (PNG)
- All drivers generate binary “waste”

SVG

- The simple case: SVG (XML)
  - XML assertions are fairly simple
  - SVG-XML still contains serialized floating point numbers
  - Platform issues, implementation changes of sprintf

SVG

- The simple case: SVG (XML)
  - XML assertions are fairly simple
  - SVG-XML still contains serialized floating point numbers
  - Platform issues, implementation changes of sprintf
- Testing schematics
  - Generate test result
  - Manual introspection
  - Store as expectation

SVG

- The simple case: SVG (XML)
  - XML assertions are fairly simple
  - SVG-XML still contains serialized floating point numbers
  - Platform issues, implementation changes of sprintf
- Testing schematics
  - Generate test result
  - Manual introspection
  - Store as expectation
- Not really “Test Driven Development” any more

PNG

- Use format specific comperator
  - imagemagick
  - pdiff (Perceptual Image Diff)
    - Tries to mimic human image perception
    - Available at: http://pdiff.sourceforge.net/
    - Not really usable for “library code”

File writing test

```
1         $command = sprintf(
2             'compare -metric MAE %s %s null:',
3             escapeshellarg( $this->filename ),
4             escapeshellarg( $other )
5         );
```

```
1         // Different versions of ImageMagick output "dB" or not
2         if ( preg_match( '/([\d.,e]+)(\s+dB)?/', $resultString, $match ) )
3         {
4             $this->difference = (int) $match[1];
5             return ( $this->difference <= $this->delta );
6         }
```

- Noise distance is quite volatile
  - GD changes output with each version
  - Stable for cairo

File writing test

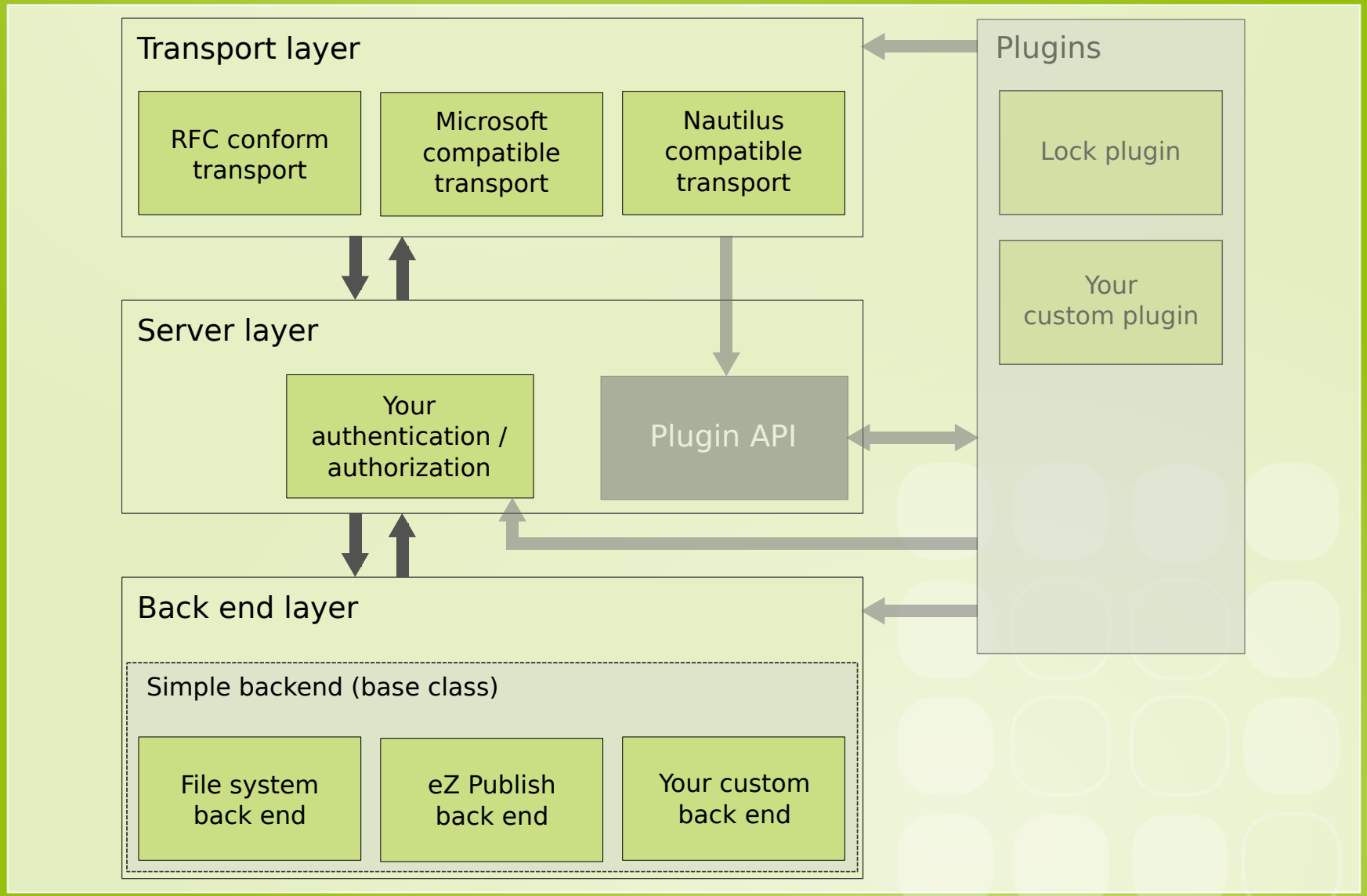
```
1         $command = sprintf(
2             'compare -metric MAE %s %s null:',
3             escapeshellarg( $this->filename ),
4             escapeshellarg( $other )
5         );
```

```
1         // Different versions of ImageMagick output "dB" or not
2         if ( preg_match( '/([\d.,e]+)(\s+dB)?/', $resultString, $match ) )
3         {
4             $this->difference = (int) $match[1];
5             return ( $this->difference <= $this->delta );
6         }
```

- Noise distance is quite volatile
  - GD changes output with each version
  - Stable for cairo

Outline

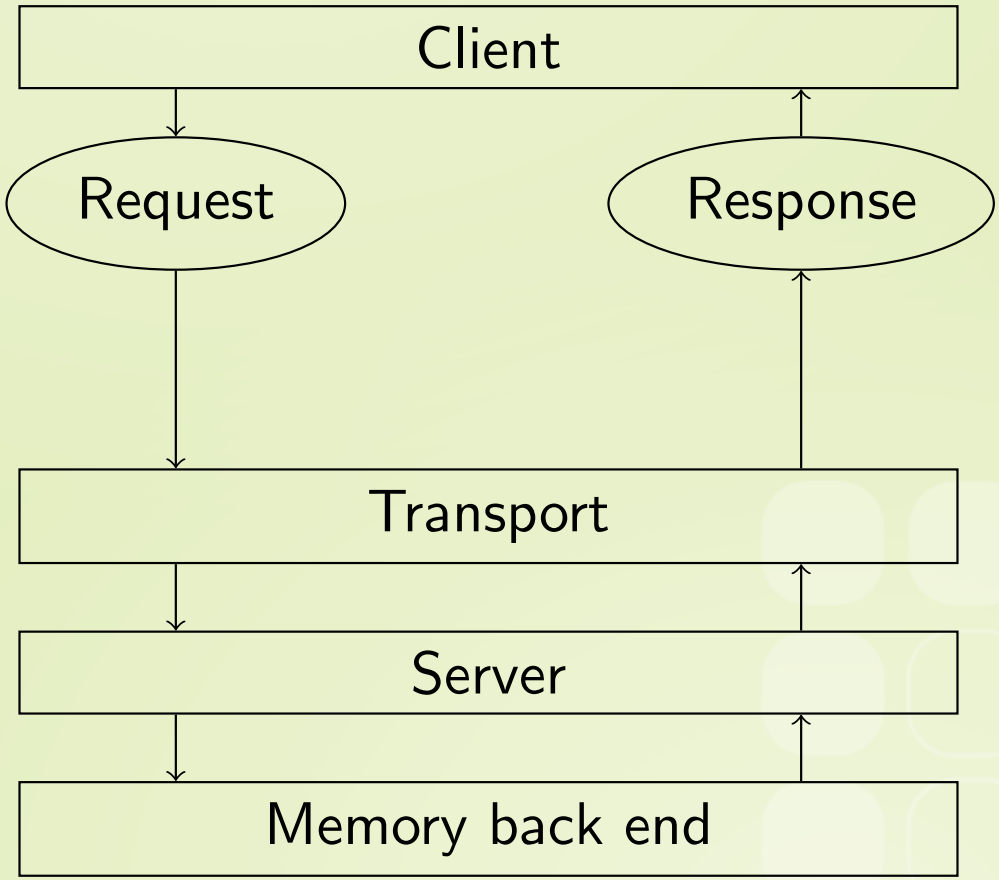
Zeta Webdav architecture



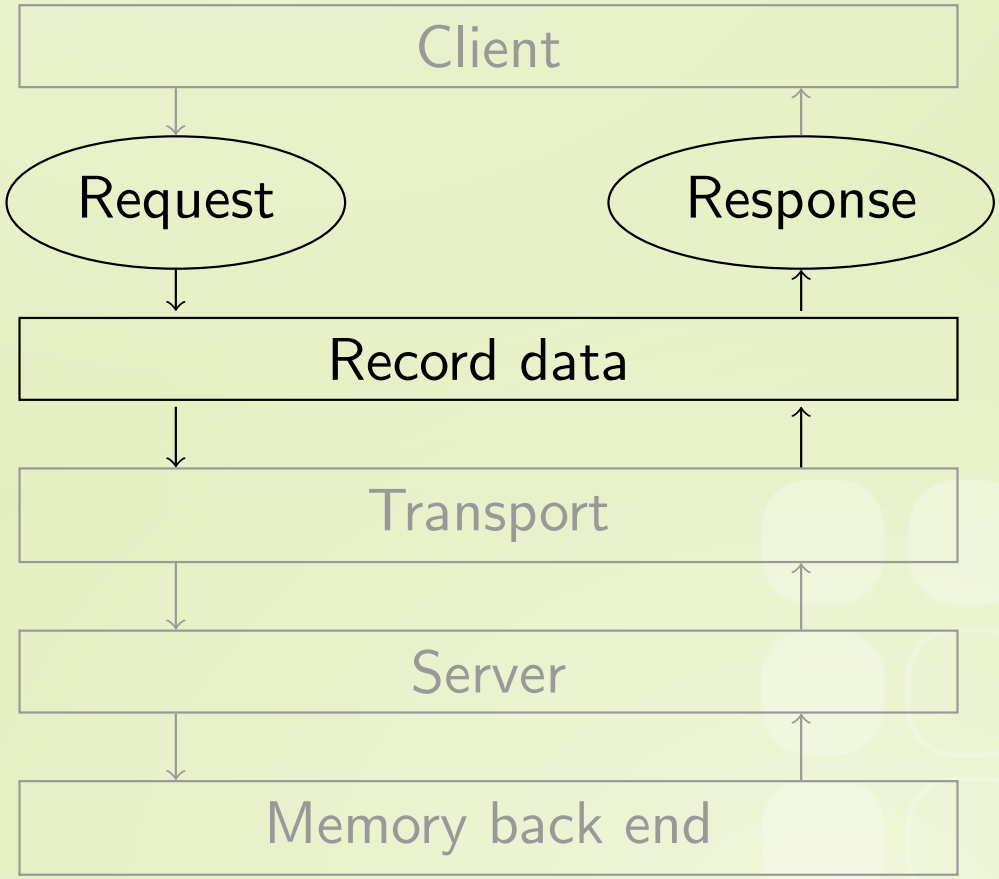
Testing a WebDAV server

- Fully unit tested
- Clients misbehave
- Abstraction to cope with such clients
- How to ensure clients work?

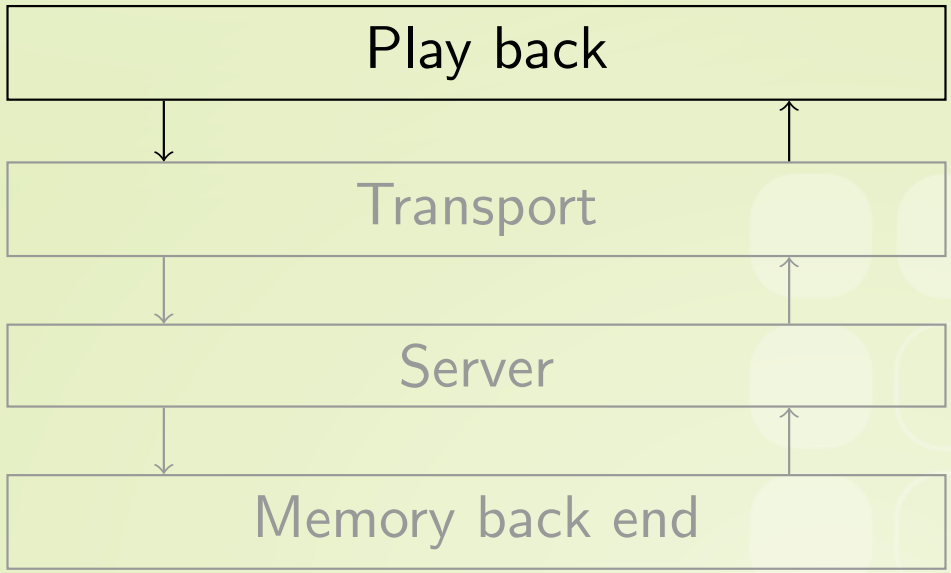
Testing a WebDAV server



Testing a WebDAV server



Testing a WebDAV server



Webdav test structure

- Request / response data store in a file
- Custom test suite to generate test sequence
- Custom test class to execute tests
- Setup class to
  - setup initial server state
  - backup and restore server state

Test suite

```
1  class ezcWebdavClientTestSuite extends PHPUnit_Framework_TestSuite
2  {
3      protected $testSets;
4      protected $setup;
5
6      public function __construct( $name, $dataFile, ezcWebdavClientTestSetup
           $setup = null )
7      {
8          $this->name = "Client: $name";
9          $this->testSets = new ezcWebdavTestSetContainer(
10             dirname( __FILE__ ) . '/' . $dataFile
11         );
12         $this->setup = (
13             $setup === null ? new ezcWebdavClientTestContinuousSetup() : $setup
14         );
15         foreach ( $this->testSets as $testId => $testData )
16         {
17             $this->addTest(
18                 new ezcWebdavClientTest(
19                     $testId,
20                     $this->setup,
21                     $testData
22                 )
23             );
24         }
25     }
26 }
```

Test suite

```
1  class ezcWebdavClientTestSuite extends PHPUnit_Framework_TestSuite
2  {
3      protected $testSets;
4      protected $setup;
5
6      public function __construct( $name, $dataFile, ezcWebdavClientTestSetup
           $setup = null )
7      {
8          $this->name = "Client: $name";
9          $this->testSets = new ezcWebdavTestSetContainer(
10             dirname( __FILE__ ) . '/' . $dataFile
11         );
12         $this->setup = (
13             $setup === null ? new ezcWebdavClientTestContinuousSetup() : $setup
14         );
15         foreach ( $this->testSets as $testId => $testData )
16         {
17             $this->addTest(
18                 new ezcWebdavClientTest(
19                     $testId,
20                     $this->setup,
21                     $testData
22                 )
23             );
24         }
25     }
26 }
```

Test suite

```
1  class ezcWebdavClientTestSuite extends PHPUnit_Framework_TestSuite
2  {
3      protected $testSets;
4      protected $setup;
5
6      public function __construct( $name, $dataFile, ezcWebdavClientTestSetup
           $setup = null )
7      {
8          $this->name = "Client: $name";
9          $this->testSets = new ezcWebdavTestSetContainer(
10             dirname( __FILE__ ) . '/' . $dataFile
11         );
12         $this->setup = (
13             $setup === null ? new ezcWebdavClientTestContinuousSetup() : $setup
14         );
15         foreach ( $this->testSets as $testId => $testData )
16         {
17             $this->addTest(
18                 new ezcWebdavClientTest(
19                     $testId,
20                     $this->setup,
21                     $testData
22                 )
23             );
24         }
25     }
26 }
```

Test suite

```
1  class ezcWebdavClientTestSuite extends PHPUnit_Framework_TestSuite
2  {
3      protected $testSets;
4      protected $setup;
5
6      public function __construct( $name, $dataFile, ezcWebdavClientTestSetup
           $setup = null )
7      {
8          $this->name = "Client: $name";
9          $this->testSets = new ezcWebdavTestSetContainer(
10             dirname( __FILE__ ) . '/' . $dataFile
11         );
12         $this->setup = (
13             $setup === null ? new ezcWebdavClientTestContinuousSetup() : $setup
14         );
15         foreach ( $this->testSets as $testId => $testData )
16         {
17             $this->addTest(
18                 new ezcWebdavClientTest(
19                     $testId,
20                     $this->setup,
21                     $testData
22                 )
23             );
24         }
25     }
26 }
```

Test case

```
1  class ezcWebdavClientTest extends ezcTestCase
2  {
3      public function __construct( $id, ezcwebdavClientTestSetup $setup, array
           $testData )
4      {
5          parent::__construct(
6              sprintf(
7                  '%s_%s',
8                  $id,
9                  $testData['request']['server']['REQUEST_METHOD']
10             )
11         );
12         $this->id = $id;
13         $this->testData = $testData;
14         $this->setup    = $setup;
15     }
```

Test case

```
1  class ezcWebdavClientTest extends ezcTestCase
2  {
3      public function __construct( $id, ezcwebdavClientTestSetup $setup, array
           $testData )
4      {
5          parent::__construct(
6              sprintf(
7                  '%s_%s',
8                  $id,
9                  $testData['request']['server']['REQUEST_METHOD']
10             )
11         );
12         $this->id = $id;
13         $this->testData = $testData;
14         $this->setup    = $setup;
15     }
```

Test case

```
1     public function setUp()
2     {
3         $this->oldLibxmlErrorSetting = libxml_use_internal_errors( true );
4
5         $this->setup->performSetup( $this, $this->id );
6
7         $this->tmpDir = $this->createTempDir(
8             get_class( $this )
9         );
10        $this->oldTimezoneSetting = date_default_timezone_get();
11        date_default_timezone_set( 'UTC' );
12    }
13    protected function tearDown()
14    {
15        libxml_use_internal_errors( $this->oldLibxmlErrorSetting );
16        $this->removeTempDir();
17        date_default_timezone_set( $this->oldTimezoneSetting );
18    }
```

Test case

```
1     public function setUp()
2     {
3         $this->oldLibxmlErrorSetting = libxml_use_internal_errors( true );
4
5         $this->setup->performSetup( $this, $this->id );
6
7         $this->tmpDir = $this->createTempDir(
8             get_class( $this )
9         );
10        $this->oldTimezoneSetting = date_default_timezone_get();
11        date_default_timezone_set( 'UTC' );
12    }
13    protected function tearDown()
14    {
15        libxml_use_internal_errors( $this->oldLibxmlErrorSetting );
16        $this->removeTempDir();
17        date_default_timezone_set( $this->oldTimezoneSetting );
18    }
```

Test case

```
1   public function setUp()
2   {
3       $this->oldLibxmlErrorSetting = libxml_use_internal_errors( true );
4
5       $this->setup->performSetup( $this, $this->id );
6
7       $this->tmpDir = $this->createTempDir(
8           get_class( $this )
9       );
10      $this->oldTimezoneSetting = date_default_timezone_get();
11      date_default_timezone_set( 'UTC' );
12  }
13  protected function tearDown()
14  {
15      libxml_use_internal_errors( $this->oldLibxmlErrorSetting );
16      $this->removeTempDir();
17      date_default_timezone_set( $this->oldTimezoneSetting );
18  }
```

Test case

```
1   public function runTest()
2   {
3       $this->setup->adjustRequest( $this->testData['request'] );
4
5       $response = $this->performTestSetRun( $this->testData['request'] );
6
7       $this->setup->adjustResponse( $response, $this->testData['response'] );
8
9       $this->assertRunCorrect(
10          $this->testData['response'],
11          $response
12      );
13  }
```

Test case

```
1   public function runTest()
2   {
3       $this->setup->adjustRequest( $this->testData['request'] );
4
5       $response = $this->performTestSetRun( $this->testData['request'] );
6
7       $this->setup->adjustResponse( $response, $this->testData['response'] );
8
9       $this->assertRunCorrect(
10          $this->testData['response'],
11          $response
12      );
13  }
```

Test case

```
1   public function runTest()
2   {
3       $this->setup->adjustRequest( $this->testData['request'] );
4
5       $response = $this->performTestSetRun( $this->testData['request'] );
6
7       $this->setup->adjustResponse( $response, $this->testData['response'] );
8
9       $this->assertRunCorrect(
10          $this->testData['response'],
11          $response
12      );
13  }
```

Test case

```
1   public function runTest()
2   {
3       $this->setup->adjustRequest( $this->testData['request'] );
4
5       $response = $this->performTestSetRun( $this->testData['request'] );
6
7       $this->setup->adjustResponse( $response, $this->testData['response'] );
8
9       $this->assertRunCorrect(
10          $this->testData['response'],
11          $response
12      );
13  }
```

Test case

```
1   protected function performTestSetRun( array $request )
2   {
3       ezcWebdavTestTransportInjector::reset();
4
5       ezcWebdavTestTransportInjector::$requestBody = $request['body'];
6       $_SERVER = $request['server'];
7
8       // ini_set( 'xdebug.collect_return', 1 );
9       // xdebug_start_trace( './traces/' . basename( $testSetName ) );
10      $this->server->handle( $this->backend );
11      // xdebug_stop_trace();
12
13      $response['status']  = ezcWebdavTestTransportInjector::$responseStatus;
14      $response['headers'] = ezcWebdavTestTransportInjector::$responseHeaders;
15      $response['body']    = ezcWebdavTestTransportInjector::$responseBody;
16
17      return $response;
18  }
19
20 }
21
22 ?>
```

Outline

PHPUnit

- Ideal for unit testing
- Flexible enough for
  - Integration tests
  - Acceptance tests
  - . . . many kinds of automated testing

Thanks for listening

Are there any questions left?

Thanks for listening

Please rate this talk:
http://joind.in/2170

Thanks for listening

Please rate this talk:
`http://joind.in/2170`

Stay in touch

- Tobias Schlitt
- toby@qafoo.com
- @tobySen
- Kore Nordmann
- kore@qafoo.com
- @koredn

Need help with PHPUnit? Rent us!
`http://qafoo.com`